CLAIMS

1. A method of creating data structures suitable for use by a virtual machine to execute computer instructions, the method comprising: converting a stream of commands and data associated with the commands into a pair of streams for use in the virtual machine, the pair of streams including a code stream that includes the commands and a data stream that includes the data associated with the commands in the code stream.

2. A method as recited in claim 1, wherein the code stream includes only commands and the data stream includes only the data associated with the commands in the code stream.

3. A method as recited in claim 1, wherein the stream that is to be converted is a Java bytecode stream, the code stream is a Java bytecode code stream that includes Java commands, and the data stream is a Java bytecode data stream that includes the data associated with the Java commands in the java bytecode code stream.

4. A method as recited in claim 1, wherein said converting of said stream comprises:

writing a representation of a first command associated with a first instruction into a code entry of the code stream;

determining whether the first command has data associated with it; and

writing a representation of the associated data or a reference to a representation of the data associated with the first command into a first data entry of the data stream when the command has an associated data.

5. A method as recited in claim 4, wherein the stream that is to be converted is a Java bytecode stream, the code stream is a Java bytecode code stream that includes Java commands, and the data stream is a Java bytecode data stream that includes the data associated with the Java commands in the java bytecode code stream.

6. A method as recited in claim 4, wherein said method further comprises:

not providing a data entry in the data stream for the command when the command does not have data associated with it.

7. A method as recited in claim 6, wherein said method further comprises:

writing a representation of second command associated with another instruction into a second code entry of the code stream;

determining whether the second command has data associated with it; and

writing a representation of the associated data or a reference to a representation of the data associated with the second command into a second data entry of the data stream when the command has associated data.

8. A method as recited in claim 7, wherein the stream that is to be converted is a Java bytecode stream, the code stream is a Java bytecode code stream that includes Java commands, and the data stream is a Java bytecode data stream that includes the data associated with the Java commands in the java bytecode code stream.

9. A method as recited in claim 8,

wherein the command and data entries can each include a number of bytecodes in the Java bytecode stream,

wherein the number of bytecodes is an integer, and

wherein each byte code can be one or more bytes.

10. A method as recited in claim 9, wherein the first and second entries of the code stream are adjacent to each other.

11. In an object oriented programming environment, a data structure for containing computer executable commands and data associated with the computer executable commands, the data structure suitable for use by a virtual machine and comprising:

a code portion having one or more computer executable commands ; and

a data stream having data corresponding to the one or more computer executable commands.

12. A data structure as recited in claim 11, wherein the code portion does not include any data associated with the computer executable commands and the data portion does not include any computer executable commands.

13. A data structure as recited in claim 11, wherein the code portion is a code stream that includes Java commands represented as bytecodes, and the data portion is a code stream that includes the data associated with the Java commands represented as bytecodes.

14. A data structure as recited in claim 13,

wherein the code stream and the data stream each comprise of a plurality of Java byte codes representing a Java bytecode stream,

wherein each Java command and data associated with that command are represented respectively in one or more bytecodes of the code stream and the data stream, and

wherein each bytecode can be one or more bytes.

15. A data structure as recited in claim 14, wherein the Java commands can be a load constant command, an invoke method command, a jump command, an instantiation command, or a get/put field command.

16. A method of executing computer instructions on a virtual machine, the method comprising:

fetching a command associated with a computer instruction from a code stream;

determining whether the command has an associated parameter;

fetching from a data stream the associated parameter of the command when said determining determines that command has an associated parameter; and

executing the command with the associated parameters after the associated parameter of the commands have been fetched.

17. A method as recited in claim 16, wherein the method further comprises:

updating a pointer to the command stream; and

updating a pointer to the data stream.

18. A method as recited in claim 16, wherein the code stream includes Java commands represented as bytecodes, and the code stream includes data associated with the Java commands represented as bytecodes.

19. A method as recited in claim 18,

wherein the code stream and the data stream each comprise of a plurality of Java byte codes representing a Java bytecode stream,

wherein each Java command and data associated with each Java command are represented respectively, in one or more bytecodes of the code stream and the data stream, and

wherein each bytecode can be one or more bytes.

20. A method as recited in claim 19, wherein the Java commands can be a load constant command, an invoke method command, a jump command, an instantiation command, or a get/put field command.

21. A method of creating data structures suitable for use by a virtual machine to execute Java instructions, the method comprising:

converting a Java bytecode into a pair of Java bytecode streams suitable for use by the virtual machine, the pair of Java bytecode streams having a Java code stream that includes Java commands and a Java data stream that includes the data associated with the commands included in the code stream

22. A method as recited in claim 21, wherein said converting of said stream comprises:

writing a representation of a first command associated with a first instruction into a code entry of the code stream;

determining whether the first command has associated data; and

reading the associated data from a Constant Pool when the command has an associated data;

processing the associated data from the Constant Pool the associated data from the Constant Pool;

writing a representation of the associated data or a reference to a representation of the data associated after said processing of the associated data;

wherein each Java command and data associated with that Java command are represented respectively in one or more bytecodes of the Java code stream and the Java data stream, and

wherein each bytecode can be one or more bytes.

23. A method as recited in claim 22, wherein said processing operates to determine a constant value associated with a Java Load Constant command.

24. A method as recited in claim 22, wherein said processing operates to determine a reference to a method invocation cell that includes information relating to a Java invoke method command.

25. A method as recited in claim 22, wherein said processing operates to determine the code stream offset and data stream offset associated with a Java Jump command.

26. A method as recited in claim 22, wherein said processing operates to process a Constant Pool associated with a Java instantiation command.

27. A method as recited in claim 22, wherein said processing operates to process a Constant Pool associated with a Java Get/Put field command.

28. A method as recited in claim 22, wherein said processing operates to process data associated with a Java load constant command, a Java invoke

writing a representation of a first command associated with a first instruction into a code entry of the code stream;

determining whether the first command has associated data; and

reading the associated data from a Constant Pool when the command has an associated data;

processing the associated data from the Constant Pool the associated data from the Constant Pool;

writing a representation of the associated data or a reference to a representation of the data associated after said processing of the associated data;

wherein each Java command and data associated with that Java command are represented respectively in one or more bytecodes of the Java code stream and the Java data stream, and

wherein each bytecode can be one or more bytes.

23. A method as recited in claim 22, wherein said processing operates to determine a constant value associated with a Java Load Constant command.

24. A method as recited in claim 22, wherein said processing operates to determine a reference to a method invocation cell that includes information relating to a Java invoke method command.

25. A method as recited in claim 22, wherein said processing operates to determine the code stream offset and data stream offset associated with a Java Jump command.

26. A method as recited in claim 22, wherein said processing operates to process a Constant Pool associated with a Java instantiation command.

27. A method as recited in claim 22, wherein said processing operates to process a Constant Pool associated with a Java Get/Put field command.

28. A method as recited in claim 22, wherein said processing operates to process data associated with a Java load constant command, a Java invoke

method command, a Java jump command, a Java instantiation command, or a Java get/put field command.